ConneXions

The Interoperability Report

October 1988 Volume 2, No. 10

ConneXions — The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published by Advanced Computing Environments, Inc., 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. Phone: 415-941-3399

© 1988
Advanced Computing Environments.
Quotation with attribution encouraged.

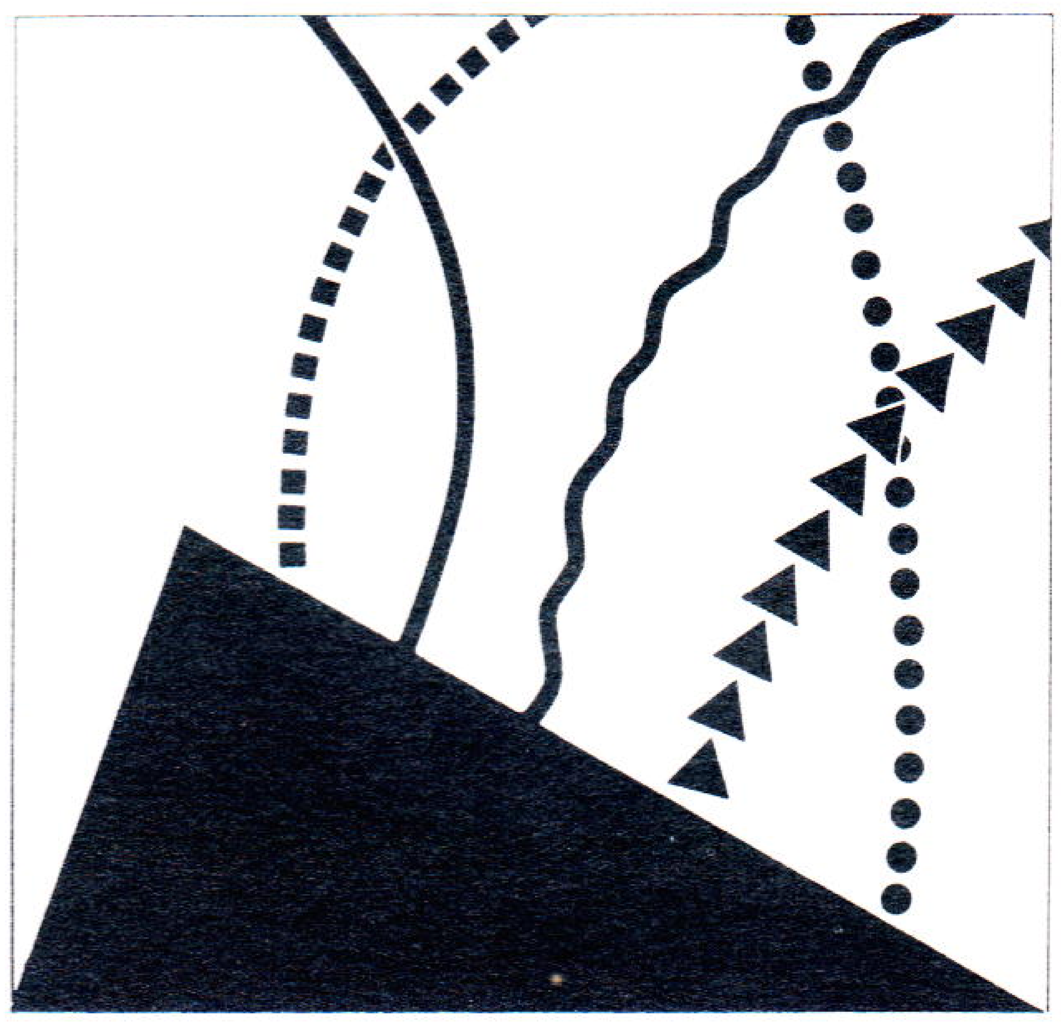
ISSN 0894-5926

From the Editor

This issue is being released at *INTEROP 88: The 3rd TCP/IP Interoperability Conference & Exhibition*, and is free to all attendees. For those readers who were unable to attend the conference, we will be bringing a full report in one of the upcoming issues. We will also be featuring articles on topics which were covered in the conference.

There are plans underway for a national research internet, sponsored by a number of federal agencies. Gregory Vaudreuil of Duke University describes this effort in an article on page 2.

TCP/IP provides interoperability at the network and transport layer and packages often include the three major applications; FTP, Telnet and electronic mail. But what about other applications such as network file systems, remote procedure calls and distributed databases? Franco Vitaliano of VXM Technologies explores the distributed applications maze and offers a possible solution with a session layer for TCP/IP in a product called TIM.



INTEROP 88

3RD TCP/IP INTEROPERABILITY CONFERENCE AND EXHIBITION

One of the main reasons TCP/IP has become so widespread is that it was supplied with Berkeley UNIX 4.2BSD and subsequently ported to a large number of systems. It is reasonable to believe that OSI protocols could be proliferated in a similar fashion. With this in mind, a number of organizations are planning to contribute OSI protocol modules to be integrated into the next BSD release. Marshall Rose of The Wollongong Group gives an overview of this project.

In an article on page 14, Jeff Mogul of DEC's Western Research Laboratory describes the BOOTP protocol which is used to boot diskless systems on LANs.

The number of vendors who support TCP/IP is steadily growing and as is customary in the "conference issue," we bring you the latest list. Be sure to let us know if you have updates to this list.

In our series of articles on the many Task Forces of the Internet Activities Board (IAB), we come to the largest one of them all: The Internet Engineering Task Force (IETF). The IETF is described here by its chair, Phill Gross.

Last month we brought you an article by Phil Karn on TCP/IP and amateur radio. We said that the amateur packet radio community is developing algorithms which are applicable to the TCP/IP community as a whole. An example is Karn's Algorithm which Phil describes under the heading "Improving Your TCP."

Advanced Computing Environments and the Corporation for Open Systems (COS) will be hosting an *OSI Product Integration Conference* at the McLean Hilton, McLean, VA, November 29—December 2 1988. Call us at 415-941-3399 for more information.

The Federal Research Internet Coordinating Committee and the National Research Network

by Gregory M. Vaudreuil, Duke University

Introduction

Computers have evolved to become a major element in modern scientific research. They are used to model real world situations too costly or impossible to test experimentally. In particular super-computers have become key elements in the emergence of computer modeling, but they have been hard to access and are very expensive. To facilitate the sharing of computing power, as well as allow the sharing of databases and exchange of electronic mail, computer communications networks are needed. The need for networking has previously been met by dedicated, single purpose networks. As computers have become more powerful, the amount of data requiring exchange has blossomed. Today's computer networks are seen as outdated and inadequate for the purposes they were designed to fulfill.

Arpanet

The first federally-financed research computer network was established by the Department of Defense. This first packet-switching network (Arpanet) was created by the Defense Advanced Research Projects Agency (DARPA) in 1969 to meet the growing needs of defense contractors for computing power. Arpanet was designed to share computing power, share databases, and provide communications services [1]. Arpanet demonstrated the initial viability of large scale packet switching, the technology underlying wide area computer networking. Packet switching is a way of exchanging digital information by breaking the data into small "packets" and sending them to the intended destination. This was a vast improvement in efficiency over point-to-point or multiplexed networks where data channels lie idle while the connected user is not sending data.

Governmental intervention is seen as necessary to unify the existing networks and achieve a national research network says Senator Albert Gore. According to a report from the Office of Science and Technology Policy (OSTP), "Other countries have recognized the value of national research networks, and following the early U.S. lead, have developed and installed national networks. As a result, these countries are now much better prepared to exploit the new distributed collaborative computing than the United States is at the present time [2]."

In 1986, concerned with the decline of American competitiveness and interested in the future role of computer networking and communications in American scientific advancement, Senator Gore submitted a bill requesting a study of the national computer networks [3]. The request, in public law 99-383, charged the OSTP to conduct; "A study of the critical problems and current and future options regarding communications networks for research computers, including super-computers, at universities and federal research facilities in the U.S."

The FCCSET report

The *Federal Coordinating Council on Science, Engineering and Technology* (FCCSET) prepared a study for the OSTP. The report finds the many existing computer networks seriously lacking in essential capacities.

Among these are several federally financed and operated networks, run by the Department of Defense, the National Science Foundation, the National Aeronautics and Space Administration, and the Department of Energy. These networks were created with the goal of meeting the computing needs of the sponsoring agencies.

Three phase project

In response to the perceived lack of a national research network, accessible to private, university, and federal researchers, the FCCSET Subcommittee on Computer Networking, Infrastructure and Digital Communications proposed a three phase project to construct a national research network, eventually to link every research institution together by high speed data links. The three phase project is outlined below.

1) Upgrade existing facilities in support of a transition plan to the new network through a cooperative effort among major government users. The current interagency collaboration to expand the Internet system originated by DARPA should be accelerated so that the networks supported by the agencies are interconnected over the next two years.

2) The nation's existing networks that support scientific research should be upgraded and expanded to achieve data communications at 1.5 Mbits/sec for 200 to 300 U.S. research institutions.

3) Develop a system architecture for a national research network to support distributive collaborative computation through a strong program of research and development. A long term program is needed to advanced the technology of computer networking in order to achieve data communication and switching capabilities to support transmission of 3Gbits/sec service with deployment within fifteen years [4].

The first recommendation of the FCCSET committee is to establish a policy of cooperation among those federal agencies with significant research networks. The current networks are loosely connected into a research Internet, allowing the limited sharing of functions, including the exchange of electronic mail.

The FRICC

In December of 1987, with the guidance of William Bostwick, DoE, five federal agencies with significant and overlapping computer networks (DARPA, DoE, HHS, NASA, and NSF), formed a cooperative and informal group to coordinate and share future network projects. The formation of the *Federal Research Internet Coordinating Committee* (FRICC) meets many of the goals of the FCCSET draft report to Congress. The primary purpose of FRICC is to share scarce government resources in a time of shrinking budgets by coordinating the existing federal networks. It is an organization that strives to save money and increase capabilities through interagency cooperation.

FRICC informally existed for many years, and was chartered in November of 1987 to formalize it's existence, William Bostwick said. Bostwick proposed the formalization of the FRICC in November of 1987. The first meeting was held in the first week of December 1987. The charter states that the organization is to coordinate the research network needs for all federal research facilities and all federally funded research.

continued on next page

National Research Network *(continued)*

According to Bostwick in a recent telephone interview, the FRICC was formed in part to provide a unified committee to work with the multinational Coordinating Council on International Research Networks (CCIRN), a committee to coordinate European networking needs. Bostwick co-chairs the CCIRN with James Hutton, the Secretary General of RARE, the French analog to the FRICC computer networking council.

The formation of FRICC in January of 1988, according to Vinton Cerf, NRI, was made easier by FCCSET's report to Congress. The FRICC, although distinct and independent of the FCCSET subcommittee on Networking, Infrastructure, and Digital Communications, has made much progress toward completion of the first phase of the National Research Internet, proposed by the FCCSET report. FRICC has enhanced the ability of federal agencies to cooperate and make efficient use of their resources. Through the Defense Research Internet, FRICC is working on the development of the technology needed to complete the National Research Internet, renewing the government's commitment to developing advanced networking technology.

Members

William Bostwick, a technology manager at the Department of Energy, is chairman of FRICC. The other committee members are Mark Pullen from DARPA, Anthony Villasenor from NASA, Stephen Wolff from NSF, John Cavallini from HHS, and Daniel Hitchcock from DoE. Many of the members of FRICC are on the FCCSET Subcommittee on Computer Networking, Infrastructure, and Digital Communications. They are Wolff, Pullen, and Cavallini.

The FRICC complements the activities of the Internet Activities Board (IAB), which is chaired by David Clark of MIT. Barry Leiner, RIACS, is the IAB liaison to the FRICC. The primary function of the IAB, according to William Bostwick, is to determine the needs of the Internet and propose technical solutions. The IAB is broken up into several *Task Forces* centered around the solution to problems in a particular area. [Ed.: See *ConneXions* Volume 1, No. 8, December 1987, and also page 20 of this month's issue.]

Resource sharing

The FRICC is dedicated to resource sharing. Long distance data transmission is one area of significant coordination. The sharing of data links greatly reduces telecommunications costs. As illustrated by Bostwick, a satellite channel to communicate with Japan and Australia costs approximately $200,000 a year. It is wasteful for each organization to finance an independent link, so the FRICC coordinates the common usage of one link. This coordination of satellite resources, is only one of several coordination projects.

RIB

Currently, the FRICC is planning the *Research Internet Backbone* (RIB), according to Vinton Cerf. The RIB is a fiber-optic trunk line that is capable of providing 1.544 Mbit/sec service on as many as 28 inter-agency links. The trunk line, consisting of a 45 Mbit/sec DS3 fiber optic channel will connect Washington, Boston, New York, Los Angeles, San Francisco, San Diego, Pittsburgh, Chicago, and Denver. The individual agencies can be linked by 1.544 Mbit/sec T1 circuits, leased digital communications lines from the telephone network, and combined into the RIB [5]. The fastest packet switches available operate at 1.5 Mbit/sec.

Both Vinton Cerf and William Bostwick note that because of excess installed fiber capacity, there have been several unsolicited bids by telecommunications concerns to provide the DS3 service, but no decision has been made at the time of this publication.

The future

The Arpanet project will be terminated. According to a DARPA memo, "DARPA is in the business of conducting research into critical *new* technologies that advance the *state of the art*. Arpanet is neither new nor state of the art. It is slow and expensive [6]." Arpanet has become a production network by default, and is no longer able to be used as a research network to try new networking concepts. Arpanet will be replaced by the *Defense Research Internet* (DRI), a new research network to develop the new 45 Mbit/sec network protocols and test new high speed switches. The DRI will be implemented on the RIB and split into two parts, a research side, to test the new technologies, and a production side.

According to Bostwick, the production channels will reduce congestion on MILNET and serve the users of the existing Arpanet. The production side will multiplex traffic from participating agencies onto the DS3 channel and drop off agency-specific T1 channels as needed. Some of these links will be operated in packet mode to support various agency-specific and inter-agency networks such as the DRI and the NSFNET.

References

[1]. Quarterman, John S. and Hoskins, Josiah C., "Notable Computer Networks," Communications of the ACM, October 1986, Volume 29, No. 10, p. 932.

[2]. "A Research and Development Strategy for High Performance Computing," Executive Office of the President, Office of Science and Technology Policy, November 1987, p. 18.

[3] "Computer Networks to Support Research in the United States," FCCSET Report to Congress: The Role of the Federal Government, September 1987, p. 189.

[4] "A Research and Development Strategy for High Performance Computing," OSTP, November 1987, p. 21.

[5] Anthes, Gary H., "Feds Band Together to Replace Nets," Federal Computer Week, March 21, 1988, p. 1.

[6] Pullen, Mark and Boesch, Brian, "Death of Arpanet and other Paranoia," (DARPA Internal Memo, Arlington, VA, March 1988).

[7] Bell, C. Gordon, "Gordon Bell Calls for a U.S. Research Network," IEEE Spectrum, February, 1988.

GREG VAUDREUIL will receive his BSE degree from Duke University in the spring of 1989. He will graduate with majors in both Computer Engineering and Public Policy Studies. He became interested in the work of the FRICC while employed at Computing Analysis Corporation.

Upper Layer Interoperability

by Franco Vitaliano, VXM Technologies, Inc.

Choices

At the present time, TCP/IP networks are in a state of transition. For some, this transition is a welcome event; for others, it is a forced march. It is ironic that TCP is seemingly reaching its zenith of acceptability just at the time that OSI is becoming a legitimate player on the network scene. So—as always seems to be the case with respect to computer technology—users are being asked to make some hard choices about future dollar investments.

When a technological cross-road such as this occurs, three things generally happen:

- Existing technologies mature and begin to show real gains in productivity.
- The newer systems offer greater benefits and promise greater flexibility.
- The computer community divides into separate camps—vendors who want to push their new wares, and users who want the maximum return on existing investments.

There are many factors preventing a clean, quick resolution of this conflict. In due course, vendors will begin to back away from their hard line "all new or nothing" stance, and reluctantly begin to offer interim solutions. Meanwhile, new technologies will begin to appear that show some real promise, both for existing and future systems.

This article talks about the old and the new, and addresses the issues in the middle. Specifically, we will look at several product trends in distributed computing and see how they might fit into the overall flow of present and future events.

NFS

Sun Microsystem's Network File System (NFS) and Remote Procedure Calls (RPCs) are a good basic starting point for a discussion. These systems, which are currently available, will most likely be a basic part of any comprehensive distributed system of the future.

NFS has become almost a de facto standard within the TCP/IP community. Apart from Sun, NFS can also be found on Apollo, Digital and IBM PC systems. NFS provides transparent user access to remote directories and files. The remote directories are incorporated into a local system tree. The net result is that a user looks out onto networks of scattered file systems as being *locally* stored. One of the benefits of NFS not talked about too much by vendors (but discovered by users) is that inexpensive third party disks can be put on the network to act as low cost servers.

RPC

An RPC is basically a mechanism for interfacing to TCP/IP without having to implement a complex, direct socket interface. An RPC utilizes a set of data conversion routines (XDR) which substitute for presentation layer functionality. An RPC resembles a local procedure call, but causes invocation of a remote process. It offers basic network functionality that is needed by most applications, such as connection establishment, data exchange, connection release and data translation—eXternal Data Representation (hence "XDR").

But, there are problems with RPCs:

- The RPC mechanism is *synchronous*, a calling program is blocked waiting for a reply. Likewise, the receiving program must find time to stop what it is doing to service the request.

- RPCs are *not* a standard, and it is open to debate what impact AT&T's UNIX Streams will have. Admittedly, unlike RPCs, Streams are specific to UNIX environments. However, methods are now being used to extend the system into other environments.

APPC versus Named Pipes

IBM's Advanced Program to Program Communications (APPC) is the mechanism chosen by Armonk as the way to go about making heterogeneous interconnections. But IBM's definition of heterogeneous means system 36s talking to 3090s; not Suns to VAXen and then on to a 3090. Whether one likes the APPC approach or not, it will have to be accounted for when planning a distributed network. The issue of APPCs will definitely have an impact on both Microsoft's and Novell's plans. Of the two companies, Microsoft and its LAN manager for OS/2 will likely have the most thinking to do about this issue. The point of confrontation is APPCs versus Named Pipes as the connection mechanism.

Last year, Microsoft and 3COM formed a joint alliance to develop local area network systems for OS/2 and LAN Manager. LAN Manager uses a convention of Named Pipes for peer to peer communications among distributed PCs. However, IBM does not endorse this pipe mechanism for PC to PC communications. Rather, it is going with APPCs. From IBM's point of view this makes more sense, as Named Pipes are almost exclusively for OS/2, whereas IBM has many types of systems to support. Thus, Big Blue's decision can only lead to 3COM and Microsoft trying to align their product strategies in line with Armonk.

Distributed operating systems

The next rung up the connectivity ladder is distributed operating systems. The primary player in this area is BBN's CRONUS distributed operating system. This system allows the development of distributed applications in a well-structured environment of objects and object managers. CRONUS provides generic functions such as global name space, global file system, and process management. Included are sophisticated automatic features for object location and operation on replicated processes. CRONUS is being utilized by SDI in its test bed program and is a well thought out system, but perhaps is still a bit early in its product maturity.

SAA

SAA (System Application Architecture) is IBM's blueprint for distributed systems architecture. The impact SAA will have on distributed systems is still being assessed. It is important to note that SAA refers to a functional spec consisting of multiple application and systems interface components, and not to a new series of tightly integrated products. An example of an SAA component is REXX, a portable command language found mostly in VM environments (Mansfield Software also makes a REXX system for IBM PCs). Simply put, SAA is not a standardized IBM operating system intended to run across all of its systems. SAA, therefore, should not be construed to be like DEC's VMS operating system.

continued on next page

Upper Layer Interoperability *(continued)*

Perhaps the only point of market comparison for SAA is X-Open, the European derived system specification that uses UNIX as its basic building block. A potential collision between X-Open and SAA may have been avoided with IBM's recent decision to join the X-Open group. The X-Open group is mostly comprised of computer vendors, such as DEC, Hewlett-Packard, NCR, and the list is growing.

Distributed database systems

At a much higher level of network functionality are distributed database systems. Marketing wise, Oracle is generally acknowledged as the leader in this area. However, Sybase is thought by some to have a superior technology. We should also mention Relational Technologies with their INGRES Star, and Rhodnius with their Empress Distributed DBMS. While both began their work in a homogeneous environment, they now exist in multi-operating environments.

Many interesting developments in distributed databases for networked PCs are also taking place, and these are likely to also impact the networking community. For example, there is the new alliance between Novell and Oracle, and a just-formed troika involving Microsoft, Sybase and Ashton-Tate. Thus, we have several major PC product players staking out their marketing turf for MS-DOS and OS/2 distributed databases.

Worth noting in this marketing shootout is which vendor's naming convention is going to be used to setup the network cross links between databases. This is really where the competitive action is, as ISVs will have to decide whose conventions they want to support.

Lastly, one should definitely not overlook IBM's upcoming database for OS/2 and their new LAN Server. Unfortunately, still too much is unknown to comment on this new product.

SQL

Given the variety and complexity of the systems listed above, it should be obvious that no new standard in distributed DBMSs is likely to emerge any time soon. The numerous combatants are too intent on fighting it out for market share. The only standard users can reasonably expect from a distributed DBMS is *SQL* (Structured Query Language), a user interface/command system for accessing distributed data.

A SQL interface can be summarized by saying that it is to distributed base access what the UNIX shell is to ordinary mortals—*cryptic*. Like UNIX, numerous front ends will certainly appear that attempt to make SQL user friendly. The question is, why didn't SQL start out that way? UNIX one can forgive, as it was designed by and for technical types. On the other hand, SQL was derived from mainframes, whose users, one would expect, are not hackers by nature.

The developments and capabilities described above give a fairly broad outline of present and future network system developments. What remains unclear, however, is the way all of these capabilities and resources will be marshalled together into a functional, standardized entity.

OSI—How we tie it all together?

Of course, OSI is considered by many to be the final outcome for computer networks. But it is generally believed that TCP/IP still has five to seven years of productive life.

As it happens, this is also about the time that OSI is expected to take before it becomes a fully developed and implemented system and methodology. So, TCP/IP users have a vested interest in preserving their investment and will want to maximize it. (By the way, we should not overlook the very large installed base of PC networks, such as Novell's, which uses a subset of Xerox's XNS as its protocol.) During this interim period, users will be looking to extend the functionality of their TCP/IP (and PC) networks to include OSI layer six and seven (Presentation and Application) functionality. At the present time, all that seems to be available for these purposes are FTP, Telnet and some E-mail systems (and NFS, if you will).

Network Management

Network Management has also become a hotly debated topic, both for TCP/IP and OSI, with various factions tugging away at each other (i.e., should vendors be trusted to implement this important function or should users define it?)

Moreover, neither OSI or TCP/IP provides for a heterogeneous peer to peer command and control system that can also operate both autonomously and intelligently. Nor does OSI or TCP/IP have a standardized programming language for creating and distributing machine independent applications across a network.

Lastly, there is also a definite need for a logical bridge between the two network technologies to facilitate the future transition.

TIM—A way to bridge the gap?

A new product has just arrived on the scene that could provide many solutions to the above TCP/IP vs. OSI debate. It can provide OSI layer five session functionality to existing TCP/IP networks, as well as be the basis for developing a layer six presentation capability. Its methods may provide a logical bridge between OSI and TCP/IP. The product is called *TIM* from VXM Technologies, Inc.

By using TIM, organizations can get rapid turn-around time for development of network or distributed applications. They avoid the wasted time and effort of trying to figure out the mysteries of TCP/IP interfacing.

Session layer

The most important ability of TIM, however, is to provide TCP/IP with an OSI session layer capability. This is presently absent from TCP/IP. NFS, FTP and Telnet are very useful, but they exist at the OSI level 7 application layer. And unlike an RPC, TIM functions provide a continuous, bidirectional, asynchronous (non-blocking) data stream between applications. TIM uses the Berkeley *socket* mechanism directly to create this stream mode between network and distributed applications.

A coherent session layer therefore enables non-blocking access to other processes on different network nodes and synchronizes their activities. This new session layer can be used as the basis for creating a presentation protocol, or for sending pure "as is" strings back and forth between applications.

VXM Technologies is planning to develop standardized TIM session functions for MAP/TOP OSI systems, as well as for the popular PC-based XNS systems such as Novell and 3COM.

continued on next page

Upper Layer Interoperability *(continued)*

It may soon be possible to have a coherent session mechanism throughout the TCP world, the PC systems market, and the future OSI networks. Such a standardized capability will greatly ease the transition from one network technology into another.

TIM-based processes can also communicate with other non-TIM based processes which use TCP/IP, so long as these other applications permit the integration of **C** language object modules into their client-server system. TIM could use this **C** interface as the point of entry. Using a standardized session functionality would permit the rapid integration of distributed systems and services. It might also free end users from being captive to proprietary TCP applications.

TIM operates in either server mode or client mode, and provides for complete control over an application's network operation. TIM includes functions which can be used use to avoid a "deadly embrace." This is a situation where both ends of the client/server connection are tied up either waiting or requesting from the other end.

Functions

The TIM software has eleven user-callable functions, among which are: create a responder (server) socket; create an initiator (client) socket; accept a connection; send and receive a message; is a message waiting; is a new connection waiting; return name of host; close current dialogue; close responder socket; close all sockets.

One can create within the application whatever type of security systems are desired. TIM provides the reliable session layer connection for the application.

The TCP/IP network functions contained in TIM are embedded in a network or distributed program. One does not need to know the intricacies of TCP/IP interface protocols. To use TIM, source code for a network or distributed program is linked with the supplied TIM **C** language object code module. This simple technique could also allow many non-networked applications to be made network capable with just minor modifications.

Aside from network session operation, TIM also provides convenient data exchange between programs running in the same computer. TIM can replace UNIX pipes or VMS mailboxes for intertask communications. A single intertask communications methodology can now be used within a single computer node, as well as across multiple, heterogeneous nodes.

Availability

The TIM software interface for TCP/IP is now available for Sun, Apollo, DEC VAXStation, VAX/VMS and ULTRIX host computers, and for PC-DOS. Other systems support is also on the way.

Conclusion

It can be concluded from the above that many exciting (and frustrating) moments await users of networked computers. But consider the alternative to this marketplace tumult—users being dominated and controlled by a slow moving, single vendor who thinks it knows what is good for them. Computer networking has finally set users free to chart their own destinies.

OSI protocols within an openly available, POSIX-conformant, Berkeley UNIX environment

by Marshall Rose, The Wollongong Group

Introduction

The DoD protocol suite, TCP/IP, enjoys much success as today's de facto solution for networking between heterogeneous systems. However, many agree that the OSI protocol suite, as promulgated by the International Organization for Standardization (ISO) will eventually dominate computer communications. Although many of the components of OSI have reached their final status by becoming full-fledged International Standards (IS), it appears that OSI will have to climb the same "power curve" in the product development cycle in order to achieve dominance.

GOSIP

One important part of the use of standards in the commercial environment is the judicious use of functional profiles which specify the core services that should be implemented. The U.S. Government OSI Profile (GOSIP) provides this focus for federal users. The aims of GOSIP are actually quite simple: it seeks to enable users to specify and procure OSI products which are interoperable, multi-vendor, and "off-the-shelf." Technically, GOSIP is aligned with the NBS OSI Implementor's Agreements. The Implementor's Agreements, which specify functional profiles of existing OSI standards, are the results of an on-going, cooperative, multi-vendor effort. In addition, the GOSIP Advanced Requirements Group provides an emphasis on those parts of OSI which are still lagging behind the needs of federal users.

GOSIP became a Federal Information Processing Standard (FIPS) in mid-August, 1988. The Department of Defense is adopting GOSIP as a co-standard with the TCP/IP suite, and intends to specify GOSIP as its standard for open networking approximately two years from now.

POSIX

In addition to networking, operating systems have also been the subject of standardization work. One such effort is POSIX, which is an IEEE proposed standard for an interface to a UNIX-like operating system. Currently, POSIX specifies only the kernel interface, but does not specify a standard method for accessing network services from the kernel interface. Work on a shell and tools standard for POSIX is also advancing.

Another U.S. Government FIPS is being developed for POSIX. It is meant to be the initial set of guidelines for procurement of operating systems for U.S. Government users.

Many have noted that the widespread use of TCP/IP was dramatically accelerated by the inclusion of these protocols in Berkeley UNIX. This modest observation leads to several questions:

- Can a reference version of OSI protocols be put into Berkeley UNIX?
- Can the Berkeley UNIX kernel be made POSIX-compliant?

continued on next page

OSI protocols *(continued)*

- Can POSIX be extended to define an interface for network services?
- Can these three goals be made openly available and ready for inclusion for the next major release of Berkeley UNIX?

The answer to all four questions is... ***yes!***

Putting the pieces together

From a distance these four goals seem somewhat ambitious: Implement a complete OSI stack. Add POSIX compliance to Berkeley UNIX. Define a networking interface from scratch. And have all of this ready for the next BSD release. However, it turns out that a large number of the pieces are already openly available. As such, the work consists mainly of filling in the gaps, integrating the components, and then testing the system.

OSI stack

To begin, an openly available implementation of the OSI upper layers already exists, the ISO Development Environment (ISODE). [Ed.: see *ConneXions*, Volume 1, No. 1, May 1987 for more information]. ISODE implements the OSI Session and Presentation layers, along with a few of the OSI Application Service Elements (such as Association Control). An implementation of the OSI File Transfer, Access and Management (FTAM) service is also included.

Other organizations are also developing the lower layers, such as ISO Transport class 4 (TP4), along with some OSI applications, such as X.400 Message Handling.

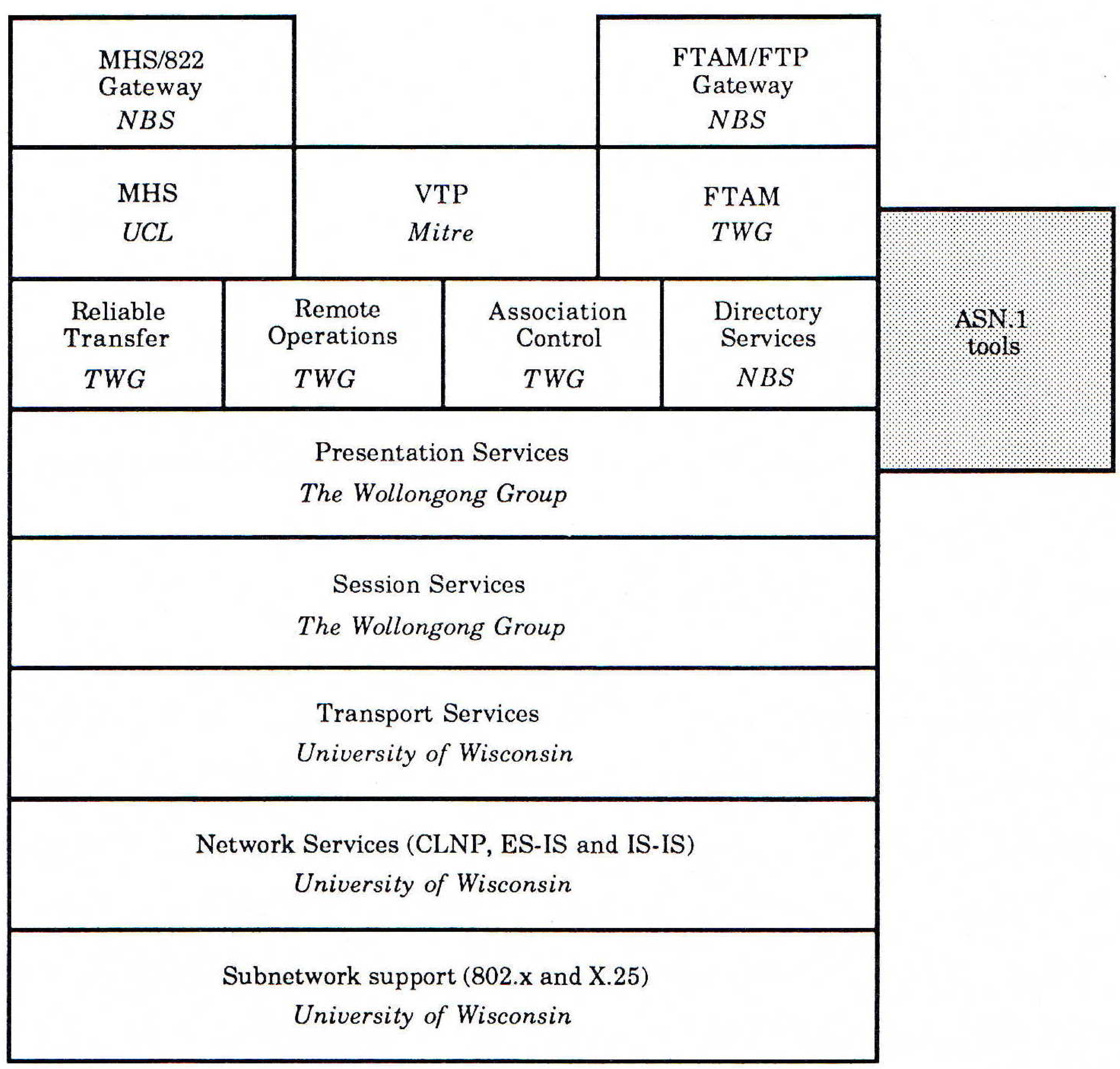


Figure 1: OSI Modules and contributing organizations

Each box in Figure 1 represents an OSI module. Most of the modules are already present in ISODE. To summarize:

- X.400 Message Handling: University College London (UCL)
- MHS/822 Gateway: The National Bureau of Standards (NBS)
- Virtual Terminal Protocol: The Mitre Corporation
- File Transfer, Access and Management: already in ISODE
- FTAM/FTP gateway: NBS
- Directory Services: NBS
- Transport Layer: joint project between the University of Wisconsin, Madison (UoW) and the IBM Corporation
- Network Layer: joint project between UoW and IBM
- Subnetwork support: joint project between UoW and IBM
- Everything else: The Wollongong Group (TWG)

Hence, the work plan is straight-forward: ISODE, which mostly implements Draft International Standards (DIS) will be upgraded to implement the final (IS) versions; other OSI applications will be integrated into ISODE; the OSINET facility will be used to perform interoperability testing; and, ultimately, conformance testing with the Corporation for Open Systems (COS) will be done. The University of California, Berkeley will integrate the OSI modules into the BSD kernel, while Wollongong will integrate modules into ISODE.

POSIX compliance

As it turns out, the actual kernel interface specified by POSIX is very close to some of the services provided by the Berkeley UNIX kernel. Hence, only minor work will be required to provide a POSIX compliant interface in Berkeley UNIX. The National Bureau of Standards is developing a conformance test suite for POSIX, and once both (independent) tasks are complete, conformance testing will be performed. Finally, as the POSIX interface becomes available in Berkeley UNIX, ISODE and the other OSI applications will be converted to use the relevant facilities.

Network Service extensions

The various parties contributing to this effort will jointly define a network service interface. Currently, a draft proposal is being advanced through the /usr/group organization. The resulting text will be submitted as a draft proposal to the POSIX committee.

Conclusions

Both OSI and POSIX will play an important role in the computer market in the years to come. By making a POSIX-compliant reference implementation openly available, we hope to accelerate the use of these standards in the short-term. The project is now underway and is expected to be completed by the end of 1989.

MARSHALL T. ROSE is a Principal Software Engineer at The Wollongong Group. He is the principal implementor of ISODE. He co-authored RFC 1006 (ISO Transport Services on top of the TCP), and was a member of the IFIP working group whose efforts led to RFC 987 (Mapping between X.400 and RFC 822). He is a member of the NSF Network Technical Advisory Group and adjunct Assistant Professor at the University of Delaware. Rose holds a Ph.D. in Information and Computer Science from the University of California, Irvine.

Booting Diskless Hosts: The BOOTP Protocol

by Jeffrey Mogul, Digital Equipment Corporation, Western Research Laboratory

What is BOOTP?

Local area networks (LANs) make it possible to use diskless hosts as workstations, gateways, terminal concentrators, etc. It is natural to bootload a diskless host over the network, and for many people, an IP-based protocol makes sense. There is a "chicken or egg" problem, however, which must be faced in order to use IP to boot a diskless host.

A diskless host, by definition, has no memory that persists across reboots, crashes, or power failures. Yet, for a host to participate in IP protocols, it must know several facts, such as its IP address and the address of a bootserver. Early diskless hosts had this information burned into PROMs, but this is inflexible and, in a large organization, a logistic nightmare. Other systems required a user to type in the addresses before booting; this is not practical for unattended systems or for naive users.

One partial solution to this problem is the use of the "Reverse Address Resolution Protocol" (RARP) [3]. RARP is straightforward, but it is not possible to implement a RARP server for 4.2BSD-derived systems without access to the kernel sources, because RARP is not itself an IP-based protocol.

The BOOTP protocol was developed partly to avoid this problem, and partly to supply additional information to the client host. A BOOTP client sends a broadcast IP packet (which it can do without knowing its own address), and receives a reply which contains its IP address, the address of a bootserver, the name of a file to bootload, and other useful information.

The BOOTP protocol is *not* used to actually bootload the file; it is used only to obtain configuration information, and another protocol must be used to transfer the bootstrap image. Normally, the TFTP protocol [9] is used for that purpose, although BOOTP does not require this.

This article is meant to be an overview of BOOTP and its use. Anyone planning to implement BOOTP should obtain RFC 951 [1], which defines the protocol, as well as RFC 1048 [8], which defines some extensions to BOOTP.

Intended environment

BOOTP relies, in its most common use, on the ability to broadcast packets on the LAN. This is reasonable, since most common LANs support broadcast or multicast. More specifically, BOOTP is intended for use on Ethernet or similar LAN technologies; in this article, I assume that an Ethernet is being used.

It is assumed that both a BOOTP server and a server for TFTP (or whatever transfer protocol is used) are available on the LAN to which a BOOTP client is attached. The BOOTP server needs a configuration file that contains a mapping between Ethernet host addresses and the corresponding IP host address; this file may contain other information as well. The TFTP server needs to have access to the boot image files, of course. Normally, both these servers will run on the same host; multiple hosts on a network may run servers, to improve availability.

In some cases, there may be no server machines on a network. The BOOTP protocol includes a mechanism that allows gateways to serve as "helpers," forwarding BOOTP requests to the appropriate server. This mechanism is covered in RFC 951, and will not be discussed in this article.

Problems solved

The BOOTP protocol may be used to solve a variety of problems. For example, a client that knows its own IP address may still need to discover the address of a potential bootserver; or, it may know both addresses but not the name of the file it is supposed to boot. The most important problem, however, is to provide a client with its own IP address.

The BOOTP protocol also includes the ability to transfer "vendor-specific" information to the client. In RFC 951, little is said about what this information might be; as a result, this feature has not been widely useful, since there is no standard interpretation of the vendor-specific information. RFC 1048 proposes standard encodings for a variety of useful information, including the subnet mask [5], the current time, the addresses of time and name servers, etc. Truly vendor-specific information can also be encoded using this scheme, without causing problems for multi-vendor networks. (It is not yet clear if RFC 1048 will be widely implemented.)

Sequence of events

A simple example may help illustrate how BOOTP is used. Imagine a diskless workstation that has just been powered on; the code resident in its ROMs follows the following sequence of events:

1. *Learn my own hardware address:* Every Ethernet host (according to the Ethernet specification [2]) must have a unique default Ethernet address; this is normally in a ROM on the Ethernet interface, and must be read by the BOOTP software.

2. *Do BOOTP protocol:* This has two phases:

   a. *Send a BOOTP request:* The request will contain the client's hardware address, and default values for other fields. Requests are sent to the default IP broadcast address, 255.255.255.255 [4]. The request should be retransmitted if no response is received; the retransmission strategy described in RFC 951 must be followed to avoid network overloads.

   b. *Receive BOOTP response:* When a response is received, the client should extract its own IP address, the IP address of the bootserver, possibly the IP address of an intervening gateway, and the name of a file to boot.

3. *Set up ARP protocol:* Now that the client knows both its hardware and IP addresses, it can handle ARP requests; it will be sending and receiving them when it starts to use TFTP.

4. *Transfer bootfile using TFTP:* The normal TFTP protocol is followed to transfer the file. The bootfile name is inserted into the TFTP Read-Request packet, which is transmitted using UDP [6] and IP in the usual way.

continued on next page

The BOOTP Protocol *(continued)*

If a gateway address was contained in the BOOTP response, then the IP packets should be sent, on the Ethernet, to that host, but in any case the packets should be *addressed*, in the IP header, to the specified server IP address. This is all the bootstrap code need know about routing.

The IP code will have to use ARP to discover the hardware address of the server (or gateway). Similarly, the client will receive ARP requests for its own address. These may arrive asynchronously; any incoming packet may be either an ARP packet or a UDP packet from the server, and code must be prepared to handle either kind.

5. *Start bootfile:* The internal format of the bootfile is not specified by BOOTP; it is up to the vendor. Often, the UNIX "`a.out`" format is used, consisting of a fixed-length header that describes the rest of the file. The bootstrap code uses this header to relocate the file, if necessary, and start it at the appropriate transfer address.

6. *Initialize normal IP network service*: Once the actual bootfile is running, it must initialize its own IP configuration. This includes its IP address, the subnet mask, and the addresses of one or more gateways. The bootstrap code can pass this information along (the subnet mask and gateway list may be present in the vendor-specific field), or the program can use protocols such as RARP and the ICMP Address Mask Request [5] to rediscover the information.

It is probably worthwhile to provide an optional interactive mode for the BOOTP client, in which it prompts the user for a filename, server IP address, or client IP address to use instead of the defaults returned by the BOOTP server. The bootstrap loader may also prompt for additional information, such as parameters to be passed to the bootloaded program.

Simplifications

Because the bootstrap code must typically fit into a relatively small amount of ROM, and because the demands of a simple bootstrap loader are less complex than those of a full-service system, a number of simplifications can be made to the protocol specifications:

- **BOOTP:** already a simple protocol, one can omit support for vendor-specific information and for the use of non-default initial values in the request packet.

- **ARP:** The ARP implementation used by the bootloader need only cache a single entry.

- **IP:** No support for IP routing is required, nor is support for IP header options (although the bootloader should not crash if options are present in received packets).

- **ICMP:** Although the ICMP specification [7] says that it "must be implemented by every IP module," it is in fact not particularly necessary or useful in a bootstrap loader.

- **UDP:** One may be tempted to leave out the creation and verification of UDP checksums, since they are officially optional, but in fact using checksums will make undetected failures (including corrupted bootfiles) much less likely. Since IP header checksums are mandatory, there will already be checksum code available; the use of a "pseudo-header" in the checksum calculation is only a minor complication.

- **TFTP:** The bootstrap loader need not implement Write Requests, nor any mode besides "octet" (binary). There is a subtle bug in the TFTP protocol that can lead to excessive retransmissions of packets; the cure is for the client to retransmit an acknowledgement packet for packet number N only when it times out waiting for packet number *N+1*, and *not* when a duplicate of packet number *N* arrives.

Server implementations

Implementation of a BOOTP server is relatively straightforward, although there are some subtleties mostly having to deal with interactions with the operating system on the server host.

For example, since the BOOTP server must send its response to a host that is not able to answer ARP requests (since it does not yet know its own IP address), the server must have a way to bypass the normal ARP mechanism. One way is for the server to update its host's ARP table, thereby making it unnecessary to actually send an ARP request to the client. A facility to do just this crept into Berkeley UNIX somewhat after the original release of 4.2BSD; if your UNIX system supports the SIOCSARP ioctl, then this facility is available. Otherwise, the response must be sent as a broadcast, which should be tolerable since it should not be sent very often.

Several versions of a BOOTP server for 4.xBSD UNIX, originally written by Bill Croft of Stanford University, are available in the public domain. You may obtain sources by anonymous FTP to SAFE.STANFORD.EDU, retrieving the file `pub/bootp.server.shar` (`pub/bootp.client.shar` may be of interest to implementors of BOOTP clients). You may also retrieve the file `pub/bootp.tar` from LANCASTER.ANDREW.CMU.EDU, be sure to use IMAGE mode since this is a binary file. Both the Stanford and CMU programs are unguaranteed, of course. A version of the server will be shipped in ULTRIX, starting with release 3.0.

Another problem is that the TFTP server implementation (`/etc/tftpd`) included in the 4.2BSD release, and therefore provided by many manufacturers, has several serious bugs. That server will work, but only if no packets are lost or damaged. Better TFTP servers are available; your vendor should provide one.

Even with the improved TFTP server, there is still a serious security problem, since UNIX TFTP server allows any host to read any "world-readable" file on your system. If your network is connected to the Internet, then you should worry about this. One solution is to modify the TFTP server to run at all times with a "restricted root," just as is done with anonymous logins in the UNIX FTP server. The `/etc/tftpd` program shipped in ULTRIX, starting with ULTRIX release 3.0, supports the use of a restricted root.

continued on next page

The BOOTP Protocol *(continued)*

Configuration management

The BOOTP server needs a configuration database, which provides the mapping between hardware addresses and IP addresses, and perhaps specifies default bootfiles for each host. If you are using a BOOTP server derived from the Stanford code, you will need to create and maintain a file called `/etc/bootptab`; here is a simplified example:

```
# Blank lines and lines beginning with '#' are ignored.
# home directory
/
# default bootfile
defaultboot
# end of first section
%%
#
# The remainder of this file contains one line per client inter-
# face with the information shown by the table headings below.
#
#host name       htype hardware addr   IP addr      bootfile
#
workstation66    1 08:00:2b:06:8e:20   130.130.1.3  unix
workstation19    1 08:00:2b:02:8a:e8   130.130.1.5  vms
lab-gateway      1 08:00:2b:02:8a:e8   130.130.1.1  gateway
```

As you can see, when you add a new host to your network, or change a host's hardware address, IP address, or default bootfile, you must update this file. If you have more than one host acting as a BOOTP server, any host listed in the configuration tables of more than one server should be listed identically.

Acknowledgements

Michael Greenwald provided me with information on solving the bug in the TFTP spec. Bill Croft and Drew Perkins wrote the 4.xBSD *bootpd* server code upon which most other servers have been based.

References

[1] Croft, W. & Gilmore, J. "Bootstrap Protocol (BOOTP)," RFC 951.

[2] "The Ethernet, A Local Area Network: Data Link Layer and Physical Layer Specifications (Ver. 1.0)," DEC, Intel, Xerox, 1980.

[3] Finlayson, R., Mann, T., Mogul, J., Theimer, M., "A Reverse Address Resolution Protocol," RFC 903.

[4] Mogul, J. "Broadcasting Internet Datagrams," RFC 919.

[5] Mogul, J. & Postel, J. "Internet Standard Subnetting Procedure," RFC 950.

[6] Postel, J. "User Datagram Protocol," RFC 768.

[7] Postel, J. "Internet Control Message Protocol," RFC 792.

[8] Prindeville, P. "BOOTP Vendor Information Protocol," RFC 1048.

[9] Sollins, K. "The TFTP Protocol (Revision 2)," RFC 783.

JEFFREY C. MOGUL received an S.B. from M.I.T. in 1979, an M.S. from Stanford in 1980, and his PhD from the Stanford Computer Science Department in 1986. While at Stanford he produced the distribution of the Stanford implementation of PUP protocol software for UNIX. Since 1986, he has been a researcher at the DEC Western Research Laboratory, working on network and operating systems issues for high-performance computer systems.

Multi-Vendor Support

The number of vendors who support TCP/IP has grown to over 180 and I am starting to have problems fitting them all on one page. (The list does *not* include the many public domain implementations of TCP/IP). Please contact me if you've got updates for the list. — Ole

ACC
Adax Inc.
ADVINTECH Corp.
Allen-Bradley
Alliant Computer Systems Corp.
Altos Computer Systems
AMD Inc.
Amdahl Corp.
Apollo Computer Inc.
Apple Computer Inc.
Applitek
Arete Systems Corp.
ARINC Research Corp.
Associated Computer Experts
AT&T
Auscom Inc.
Aydin Monitor Systems
Banyan Systems Inc.
BBN Communications Corp.
BBN Laboratories
BDM
Beame & Whiteside Software Ltd.
BICC Networks
Black Box Corp.
Britton-Lee Inc.
Bull AG
Butler & Curless
Canaan Computer
Cadtronic
Celerity Computing
Charles River Data Systems
Chi Corp.
Chipcom
cisco Systems Inc.
Claflin & Clayton Inc.
ComConsult
Communication Machinery Corp.
Computer Network Technology
Concurrent Computing
Control Data Corp.
Convergent Technologies
CONVEX Computer Corp.
Computer Sciences Corp.
Counterpoint Computers
Cray Research Inc.
Cydrome
Dana Group
DANET GmbH
Data General
Datapoint Corp.
DevelCon
Digital Equipment Corp.
Eastman Communications
ELXSI Inc.
Encore Computer Corp.
Eon Systems Inc.
Epilogue Technology Corp.
Excelan Inc.
Failsafe Computer Systems Inc.
Fibronics International Inc.
Ford Aerospace & Communications Corp.
Frontier Technologies Corp.
FTP Software Inc.
Gateway Communications
GE Calma
GEI Rechnersysteme GmbH
General Electric Company
General Instruments
GigaMos Systems
Gold Hill Computers
Gould Inc.
Grasshopper Group
GTE
Halley Systems
Harris Computer Systems Division
Hemispheres Hi-Tech
Hewlett-Packard Company
Honeywell Information Systems
IBM
IMAGEN Corp.
Informix Software
Intel Corp.
Interactive Systems
Interfirm Graphic Systems
Integrated Solutions Inc.
Intergraph Corp.
Intermetrics
Internet Systems Corp.
Interphase Corp.
Kinetics Inc.
KMW Systems
D. Köpke RzK
Lachman Associates
LANEX
Learning Tree Software Inc.
Little Machine Inc.
Locus
Marble Associates Inc.
MARI Advanced Microelectronics Ltd.
Masscomp
Maxim Technologies
MICOM-Interlan
Microport Systems Inc.
MIPS
Mitek Systems Corp.
Mitre Corp.
Motorola Inc.
Mt Xinu
NBI Inc.
NCR Corp.
NCR Comten
Nestar
Network General Corp.
Network Research Corp.
Network Solutions Inc.
Network Strategies Inc.
Network Systems Corp.
Nixdorf Computer Corp.
Norsk Data A/S
Novell
NYSERNet
Opus Systems
Oracle Corp.
Panda Programming
PCS Computer Systeme GmbH
Phoenix Technology
Plexus Computers Inc.
Positronica
Prime Computer Inc.
Process Software Corp.
Proteon Inc.
Protocom Devices
Pyramid Technology Corp.
Relational Technology Inc.
Research Equipment
Research Triangle Institute
Ridge Computers
SAIC
Samsung Semiconductor & Telecommunications Co. Ltd.
Santa Cruz Operation Inc.
SCI Technology Inc.
Schlumberger/Applicon
Schneider & Koch
SCOPE Inc.
Sequent
SIEMENS
Silicon Graphics
Simware Inc.
Sirius Systems Inc.
SMS Data Products Group
Software Kinetics Ltd.
Software Systems Associates
SPARTA
Spider Systems Ltd.
SRI International
Stemmer Elektronik
Sterling Software
Stollmann GmbH
Stride Micro
Sun Microsystems Inc.
SYNELEC Datensysteme GmbH
Symbolics Inc.
SynOptics Communications
Syntax Systems
Systems Strategies
Sytek Inc.
Tandem Computers Inc.
Telemation
Televideo
Tektronix Inc.
Texas Instruments Inc.
TGV
THG
3Com
TOPS
TRW
Tracor
Ungermann-Bass Inc.
UNIQ Digital Technologies
UniSoft
Unisys Corp.
Univation
Valid Logic
Vitalink Communications Corp.
VXM Technologies Inc.
Wang Laboratories Inc.
Wellfleet Communications Inc.
Wetronic Automation GmbH
Western Digital
The Wollongong Group
Xerox Corp.
Xios Systems
Xyplex

The Internet Engineering Task Force

by Phill Gross

History

DARPA originally formed the Internet Activities Board (IAB) to oversee and coordinate work under DARPA's Internet Research Program. It was decided to organize *Task Forces* (TFs) under the IAB in various areas of interest, such as Gateway Algorithms, End-to-End Services, and Privacy. The chairs of these Task Forces together would compose the core of the IAB itself. The IAB is chaired by Dr. David Clark, MIT and the vice-chair is Dr. Jon Postel, ISI. In keeping with their mission of overseeing the Internet Research Program, the TFs all had a distinctly research-oriented flavor. It wasn't long, however, before the IAB's success caught up with it. I certainly don't need to relate here how the Internet has grown in both size and importance in recent years. Along with this success came a subtle shift in the IAB's responsibilities.

Evolution

As recounted in *ConneXions*, Volume 1, No. 8, December 1987, the IAB has grown into the coordinating committee of the Internet. In addition, the original DARPA emphasis has broadened out to include other agencies, such as DCA and NSF. Even more recently, five agencies (DARPA, DoE, HHS, NASA, and NSF) have joined together to form the Federal Research Internet Coordinating Committee (FRICC). Their goal is to pursue the networking recommendations laid out by the Federal Coordinating Committee on Science, Engineering and Technology (FCCSET). These recommendations include building a high-speed national network within five years. Thus, the IAB now has wide agency involvement and has established liaisons with other R&D bodies. Throughout all this growth and evolution, the TFs of the IAB have been successful in maintaining a forward-looking research orientation.

However, one Task Force was particularly hard hit by the growth in the Internet. The original mission of the Gateway Algorithms and Data Structures Task Force (GADS), chaired by Dr. David Mills (University of Delaware), was to pursue research topics at the Internet layer, such as Internet routing. Even at its early meetings, the group found itself diverted from advanced topics to deal with more operational problems like how to adapt EGP for a large, diverse, operational Internet. It soon became clear that another group was needed with a nearer-term focus. At the IAB meeting in January 1986, it was decided to divide the GADS group into the Internet Architecture Task Force (INARC) and the Internet Engineering Task Force (IETF). The Architecture TF maintained the original research mission with Dave Mills as chair, while the IETF absorbed the nearer-term engineering and technology transfer responsibilities. Mike Corrigan (OSD, then Technical Program Manager for DDN) was the original chair of the IETF.

IETF Charter

The IETF was given the charter to identify and coordinate solutions for problems in the management, engineering, and operations of the Internet. Furthermore, the IETF would act to identify problems in the Internet protocol architecture, recommend near-term and mid-term approaches, and help coordinate prototyping efforts. In summary, the IETF would not be concerned with longer-term pure research topics, concentrating instead on nearer-term operational issues and mid-term R&D issues that would have a high operational payoff without significant changes to the basic architecture.

Second evolution

At first, the IETF meetings felt very much like the old GADS meetings, since many of the issues and participants were the same. With continued growth, in particular with the advent of the original NSFNET, the IETF began to develop a much larger constituency. Gateway vendors for the various regional networks began to attend. Finally in March 1987, the NSFNET Routing Group joined forces with the IETF, approximately doubling its size at that time, and the format was reorganized to take on its current structure. [Ed.: see *ConneXions*, Volume 1, No. 2, June 1987]. When Mike Corrigan moved to the Office of the Assistant Secretary of Defense (OASD), the IAB asked the author to chair the group. By now, the IETF has grown into a forum for government agencies, contractors, university researchers, vendors, network users, and industry R&D labs.

New format

After the NSFNET group joined, IETF meetings essentially became open forums. With this increased size and scope, it was necessary for the IETF to develop a structure of its own. Specific problem areas within the IETF are now addressed in "Working Groups." There were originally three Working Groups—Routing, Domains, and Performance. In yet another measure of growth, there are now 18. Five separate groups grew out of the original routing group. These include efforts to improve EGP, to investigate a successor to EGP, to prototype an open IGP based on the proposed ANSI routing standard, and a routing advisory group for operational networks.

Other important IETF Working Groups that have received attention lately include Internet Host Requirements (an effort chaired by Bob Braden of ISI to define standard host protocol operations; a host analog to RFC 1009), CMIP-Over-TCP (CMOT) [or "Netman"] (chaired by Lee LaBarre of Mitre; to specify and prototype the use of ISO CMIP for Network Management within the Internet architecture), SNMP Extensions (chaired by Marshall Rose of TWG; to explore and define possible extensions to SNMP, as specified in RFC 1052), Internet Management Information Base (MIB) (chaired by Craig Partridge of BBN; to define an Internet standard MIB for both CMOT and SNMP), and Serial Line IP (SLIP) (with three co-chairs; to specify the standard usage of IP over serial links).

Many of the early Working Groups were established by the IETF chair. More and more, however, interested parties are asking to form Working Groups under the auspices of the IETF. The SLIP and CMOT groups fall into this category, as do other groups on Public Data Network (PDN) routing, Telnet, and Internet authentication. It is probably fair to consider the RFC 1052-spawned groups in this category, too. This is a new and promising trend and it indicates to me that activities under the IETF are taking on a life of their own.

For those who feel they have an issue that might be profitably addressed in this way, the rules for starting a Working Group are quite simple. First, the organizer must write a fairly narrow charter stating the goals and expected duration of the group. This will help both to justify the group and to measure its progress. Then, collect a reasonably small set of active core participants and carry on electronic mail discussions or hold separate meetings to work on the issue. Finally, produce an RFC or working paper (IDEA) to document the effort. Working Groups are also expected to report their progress at the IETF plenary meetings and to contribute written status reports to the IETF Proceedings.

continued on next page

The Internet Engineering Task Force *(continued)*

IETF plenary meetings

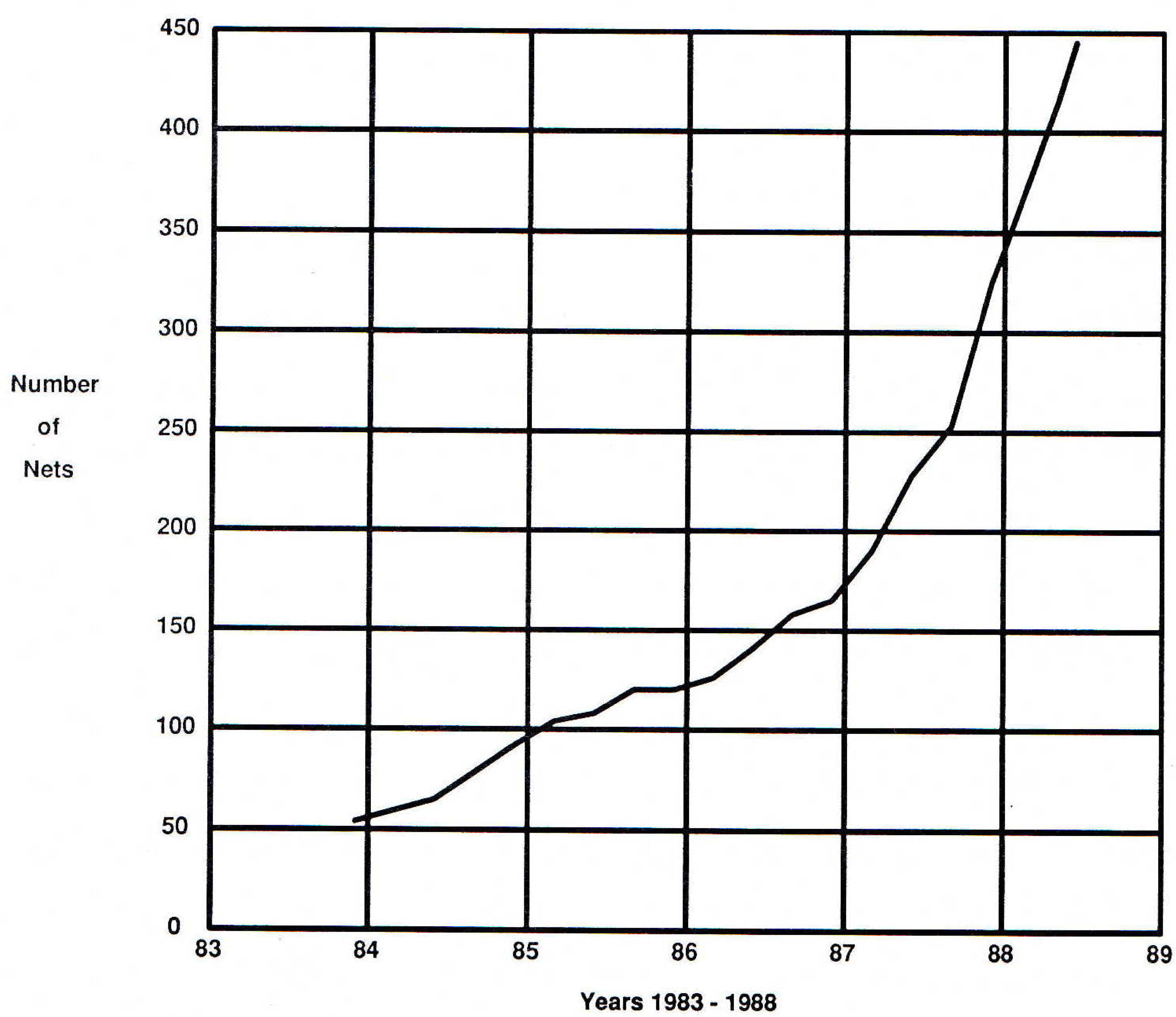
The IETF holds plenary meetings four times a year. The Plenaries are three days long, with the first day and the morning of the second day devoted to Working Group breakout sessions. The remainder of the time is divided between network and agency status reports, Working Group reports, and technical presentations. Proceedings are produced for each meeting, which include minutes of the plenary sessions, written Working Group reports, and all presentation foils. The Network Information Center (NIC) at SRI has recently begun to act as a distribution point for IETF Proceedings.

The last IETF meeting was in June at the US Naval Academy in Annapolis, where 13 working groups met and reported. One highlight was a report by Hans-Werner Braun of Merit, on the status of the new NSFNET. The report was so glowing that Hans-Werner volunteered to host the next IETF meeting in October at the University of Michigan and demonstrate the operation of the new NSFNET.

Mailing lists and additional information

There are two main IETF mailing lists: ietf-tf@venera.isi.edu and ietf-interest@venera.isi.edu. To join the interest list, send a request directly to westine@isi.edu. To join the main IETF list, send a note to gross@mitre.org, with a brief statement explaining your interest in the IETF. It is worth noting that many of the Working Groups now have mailing lists of their own. For additional information on the IETF or any of its Working Groups, drop a note to gross@mitre.org.

PHILL GROSS is a Member of the Technical Staff at the Mitre Corporation, where he is involved in DoD protocol engineering, protocol performance and DoD OSI transition planning. He is chair of the IETF. His MS in Computer Science is from The Pennsylvania State University.



The Internet has been growing at an impressive rate. Mike Brescia of BBN presented this graph at the last IETF meeting.

Improving Your TCP: "Karn's Algorithm"

by Phil Karn, KA9Q

RTT

Efficient TCP performance depends on the sender having an accurate, up-to-date estimate of the *network round trip time* (RTT). It is essential that TCP track changing RTTs as quickly as possible, but unfortunately the retransmissions caused by timeouts greatly complicate the job.

Retransmission ambiguity

When a retransmission timeout occurs, it may be "real," i.e., caused by actual data or acknowledgement (ACK) packet loss, or "spurious," i.e., caused by a sudden increase in RTT (or an incorrect initial estimate). Unfortunately, when a packet has been retransmitted a TCP ACK does not indicate which transmission it is responding to, so the sender is unsure of the true RTT for that packet. We call this the "retransmission ambiguity" problem.

If the sender incorrectly assumes that a timeout is spurious and measures RTT from the first transmission, then the measurement will include its own retransmission timeout interval and be erroneously large. Even a moderate network packet loss rate can cause an unrealistically *high* estimated RTT and poor throughput.

Conversely, if the sender assumes that a timeout was real when it was actually spurious and therefore measures RTT from the last transmission, it will obtain an artificially *low* value. The sender may retransmit packets repeatedly even though none are actually being lost. Once again, poor throughput results. Even worse, Internet bandwidth is being wasted.

Two-part solution

The first part of my solution to this problem is to reject all RTT measurements based on retransmitted packets. They are inherently unreliable, and cannot be made reliable without changing TCP. But without fresh RTT measurements, the sender's estimated network RTT cannot be updated and spurious timeouts and retransmissions might continue indefinitely. Something that guarantees new and accurate RTT measurements is also needed.

The second part of my solution involves changing the way the retransmission timer is set when timeouts occur. Normally it is increased ("backed off") on each consecutive retransmission and returned to its original value (a function of the sender's RTT estimate) when an ACK is finally received. My solution instead calls for "clamping" (holding) the retransmission timer at the backed-off setting while the *next* data packet is sent. It is restored to its normal value only if that packet is ACKed without a timeout and retransmission; otherwise the retransmission timer is backed off even further. This technique guarantees that even when the network RTT is increasing rapidly, the retransmission timer eventually reaches an interval that permits a RTT measurement to be made without "contaminating" it with a spurious retransmission. These measurements then update the sender's estimate, driving it toward the correct value. Although it's still possible for subsequent spurious retransmissions to occur (depending on the response time of the particular smoothing algorithm in use) the estimate will eventually reach the correct new value, usually after several packets.

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1

CONNEXIONS

PUBLISHER Daniel C. Lynch

EDITOR Ole J. Jacobsen

EDITORIAL ADVISORY BOARD Dr. Vinton G. Cerf, Vice President, National Research Initiatives.

Dr. David D. Clark, The Internet Architect, Massachusetts Institute of Technology.

Dr. David L. Mills, NSFnet Technical Advisor; Professor, University of Delaware.

Dr. Jonathan B. Postel, Assistant Internet Architect, Internet Activities Board; Division Director, University of Southern California Information Sciences Institute.

Subscribe to CONNEXIONS

U.S./Canada **$100. for 12 issues/year**
International **$ 50. additional per year**

Name ______________________ Title ______________________

Company ______________________

Address ______________________

City ______________________ State __________ Zip __________

Country ______________________ Telephone (   ) ______________________

☐ Check enclosed (in U.S. dollars made payable to CONNEXIONS). ☐ Bill me/PO# ______________
☐ Charge my ☐ Visa ☐ Master Card Card # ______________________ Exp. Date __________

Signature ______________________

Please return this application with payment to:
Back issues available upon request $10./each

CONNEXIONS
480 San Antonio Road Suite 100
Mountain View, CA 94040
415-941-3399